Is Transition Systems Approach of Modeling Software Systems Hard to Learn and Use?

Mrityunjay Kumar
IIIT Hyderabad
India
mrityunjay.k@research.iiit.ac.in

Venkatesh Choppella
IIIT Hyderabad
India
venkatesh.choppella@iiit.ac.in

ABSTRACT

Modeling software systems as transition systems can improve system comprehension for novice engineers and graduating students. However, this requires them to learn the vocabulary of transition systems and its use. We hypothesize that it is not hard for them to do so. To test this, we did a very short intervention in a software engineering course for two cohorts of students. Based on the transition systems knowledge demonstrated by the cohorts, we conclude that it is not hard for them to learn the vocabulary of transition system and its use. This result gives confidence to start designing longer intervention and study their effectiveness.

CCS CONCEPTS

• **Computing methodologies** → *Modeling methodologies*; • **Social and professional topics** → **Software engineering education**.

KEYWORDS

transition systems, system modeling, novice engineers

ACM Reference Format:
Mrityunjay Kumar and Venkatesh Choppella. 2023. Is Transition Systems Approach of Modeling Software Systems Hard to Learn and Use?. In *Proceedings of the ACM Conference on Global Computing Education Vol 2 (CompEd 2023), December 5–9, 2023, Hyderabad, India.* ACM, New York, NY, USA, 1 page. https://doi.org/10.1145/3617650.3624931

1 INTRODUCTION

A system can be modeled using a Transition Systems formalism (for example [3]). We model a **Transition System** for a software system as a six tuple [1] $\{X, X^0, U, f, Y, h\}$ where X is the set of states, X^0 is the set of initial states, U is the set of actions, f is the transition relation which is a subset of $(X \times U) \times X$, Y is the set of observables (or output space), and h is a *display map*, mapping states to observables, $y = h(x)$. A transition system models the dynamics of the software system.

Software Systems are getting complex over time. At any product company, it is becoming harder for the novice engineers to understand the products in the limited time available for onboarding. In [2], we suggest that modeling new features using transition systems [1] can help them design and understand systems easily.

CompEd 2023, December 5–9, 2023, Hyderabad, India

ACM ISBN 979-8-4007-0374-4/23/12.
https://doi.org/10.1145/3617650.3624931

However, it begs the question: how hard it is for CS students to learn and use transition systems? To understand this, we designed an experiment with two cohorts using a light intervention. **Cohort A (N=9)** consisted of undergraduate second year CS students from a leading liberal arts college. **Cohort B (N=27)** consisted of undergraduate third and fourth year students, and equal number of Masters students who worked in groups of five (there were 27 groups).

The intervention was done as part of their Software Engineering course. It consisted of: 1) one lecture of 90-120 minutes (Online for Cohort A, in-person for Cohort B), 2) 5 page lecture notes with basics and examples, 3) an assignment to model given system(s) using transition system. Cohort A was given a simple design problem and a toy system to model; Cohort B was given a real system (parts of an existing appointment booking system) to model, given they had done more CS courses and were working in groups.

For the purpose of this poster, the assignment submissions were re-evaluated for demonstrated competencies around identification and representation of a transition system model and its components: 1) State spaces (X and X^0), 2) Action space (U), 3) Transition Function (f), 4) Observable and display map (Y and h), and 5) the six-tuple being an acceptable model of the system. Scores were given as 0 (does not demonstrate), 10 (partially demonstrates) and 20 (fully demonstrates).

Average score for each of the cohorts was 75% or more. Surprisingly, even Cohort A did well even though they had much less CS education than Cohort B. The cohorts didn't have difficulty in understanding and applying basic concepts of transition systems. Most of the challenges faced by students were in using set theoretic notations (they resorted to writing imperative program fragments instead) and distinguishing between transition function f and display map h. These suggest the need to focus on teaching some basic discrete mathematics when teaching transition systems. Subsequent interactions with the students suggested that their familiarity with theory of computation and state machines helped them in understanding and using transition systems easily.

Our future work will be focused on designing and testing large interventions (workshops and half-semester courses), evaluating how this modeling aids in system comprehension, and introducing transition systems in other computer science courses.

REFERENCES

[1] Venkatesh Choppella, Kasturi Viswanath, and Mrityunjay Kumar. 2021. Algodynamics: algorithms as systems. In *2021 IEEE Frontiers in Education Conference (FIE)*. IEEE, 1–9.
[2] Mrityunjay Kumar and Venkatesh Choppella. 2023. A modeling language for novice engineers to design well at saas product companies. In *Proceedings of the 16th Innovations in Software Engineering Conference*, 1–5.
[3] Bernard P. Zeigler. 1989. DEVS representation of dynamical systems: Event-based intelligent control. *Proceedings of the IEEE*, 77, 1, 72–80.